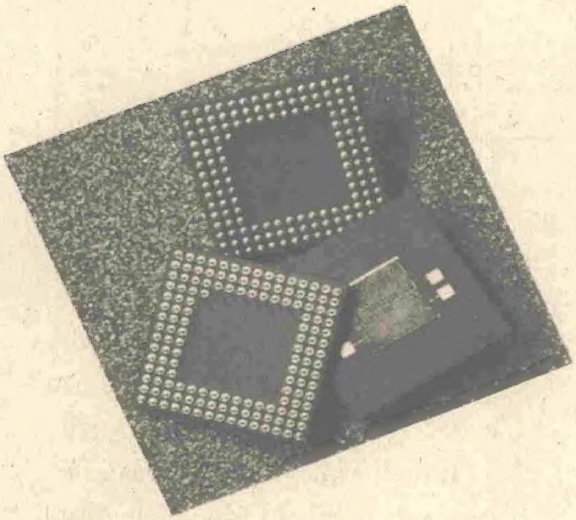
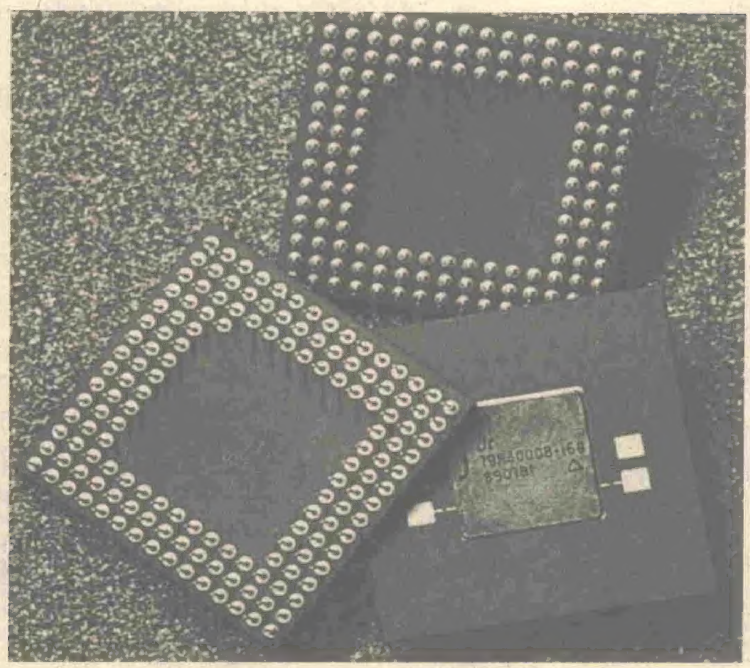
เอนก ศิรจริยาภิวัฒน์

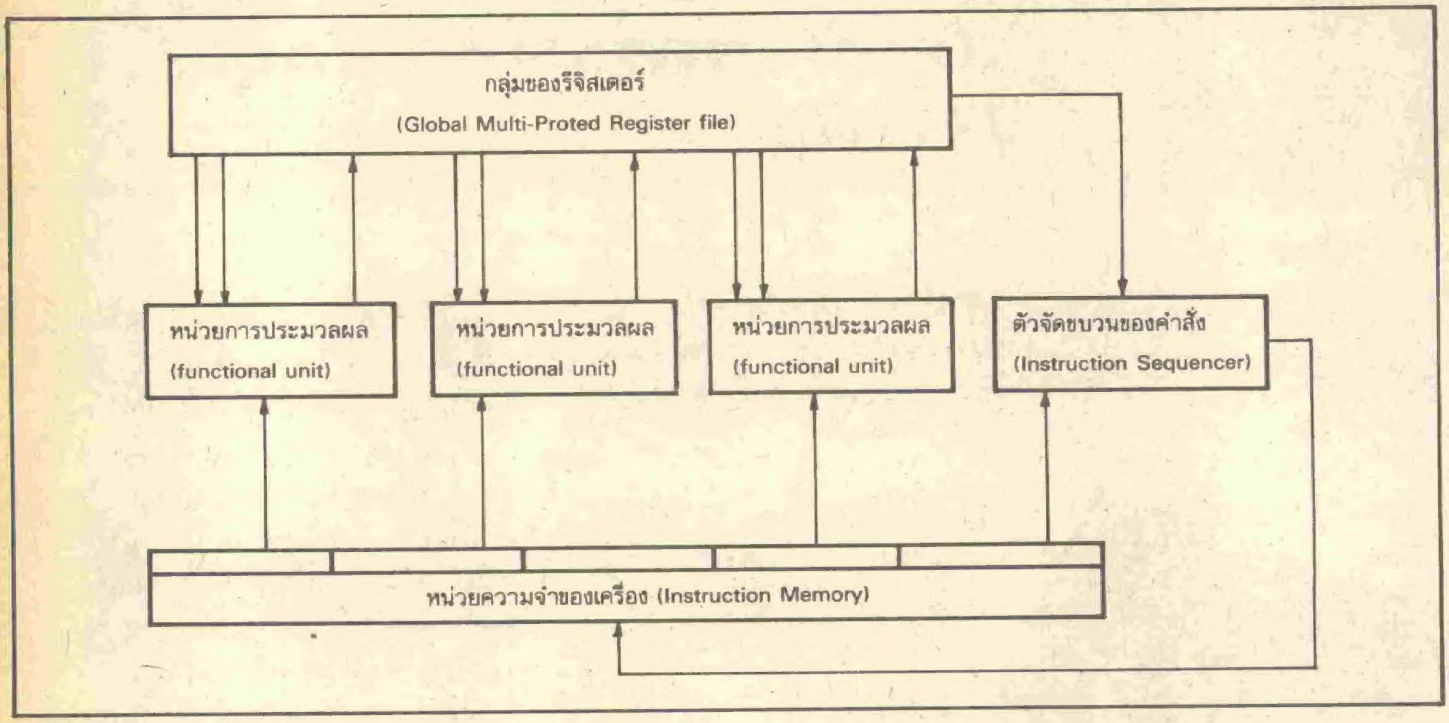
VLIW
สถาปัตยกรรมที่ใช้กับ RISC

สถาปัตยกรรมแบบใหม่ ซึ่งนำมาใช้กับการประมวลผลแบบขนานของซีพียู RISC





สำหรับเราบางคนการอ่านหนังสือและดูทีวีไปพร้อมกันจะกระทำได้ยากและเกิดปัญหาขึ้น เช่นเดียวกันกับการทำงานของคอมพิวเตอร์แบบขนาน ถ้าทำงาน 2 งานพร้อมกัน และงานทั้งสองต้องทำไปร่วมกันจะทำให้การประมวลผลไม่มีประสิทธิภาพเพียงพอ เพราะงานหนึ่งต้องรอผลจากอีกงานหนึ่ง ซึ่งทำให้การทำงานไม่ได้เป็นไปแบบขนานอย่างแท้จริง ด้วยสถาปัตยกรรมแบบใหม่ ซึ่งเรียกว่า Very Long Instruction Word (VLIW) สามารถนำมาแก้ปัญหาดังกล่าวได้ โดยใช้หลักการง่าย ๆ ว่า ถ้าใช้หน่วยประมวลผลตัวเดียวทำงานได้เร็ว ตัวประมวลผล 2 ตัวย่อมทำงานได้เร็วกว่า และถ้าเป็นตัวประมวลตัวเดียวแต่ใช่ว่าจะทำให้การทำงานของโปรแกรมเร็วขึ้นเป็น 2 เท่าเสมอไป เพราะการทำให้การประมวลผลเร็วขึ้น 2 เท่า ไม่ง่ายเหมือนกับการประกอบตัวประมวลผล 2 ตัวลงกล่อง ตัวประมวลผลทั้งสองตัวจำเป็นต้องทำงานให้ได้จังหวะ (synchronize) ซึ่งกันและกันด้วย ถ้าเราสามารถแบ่งงานออกเป็น n ส่วนได้ โดยแต่ละส่วนไม่ต้องทำงานร่วมกับส่วนอื่น จะทำให้การทำงานของตัวประมวลผล n ตัวมีความเร็วเกือบเท่ากับ n เท่าตัวของการใช้ตัวประมวลแบบตัวเดียว แต่ในทางตรงกันข้ามถ้างานมีความยุ่งยากขึ้น งานแต่ละงานต้องรอผลของอีกงานหนึ่ง จึงต้องมีการติดต่อสื่อสารและติดตาม



รูปที่ 1 โครงสร้างของ VLIW machine

(track) ระหว่างงานแต่ละงานนั้น ซึ่งเป็นคอขวด (bottleneck) สำหรับการทำงานแบบขนาน สถาปัตยกรรมแบบ VLIW จะสร้างตัวประมวลผลขนาดใหญ่ที่หน่วยประมวลผลคณิตศาสตร์ n หน่วย ซึ่งต่ออยู่กับส่วนของรีจิสเตอร์กลุ่มเดียว คำสั่ง Very Long Instruction Word : VLIW มาจากความจริงที่หน่วยคณิตศาสตร์ทั้ง n หน่วยจะรู้ว่าตัวเองมีหน้าที่อะไรจะต้องทำ ซึ่งทำให้คำสั่งนั้นมีขนาดความยาวเป็น n เท่าของคำสั่งปกตินั่นเอง

คำสั่งแยกทิศทาง

เรามาดูตัวอย่างการประยุกต์ใช้ VLIW สักตัวอย่างหนึ่ง โดยสมมติการทำงานโดยมีการแยกทิศทาง (branch) ของการทำงานดังนี้ สมมติว่าก่อนถึงคำสั่ง Branch คอมพิวเตอร์จะกระทำคำสั่งกับรีจิสเตอร์ดังนี้

$R1 + R2 \rightarrow R1$

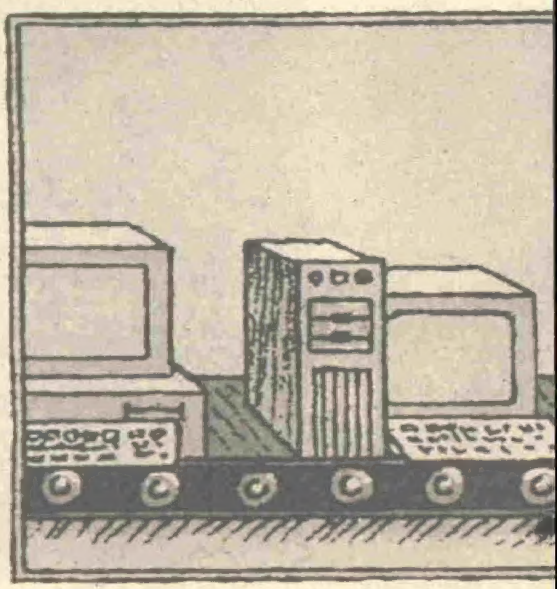
$R3 + R3 \rightarrow R3$

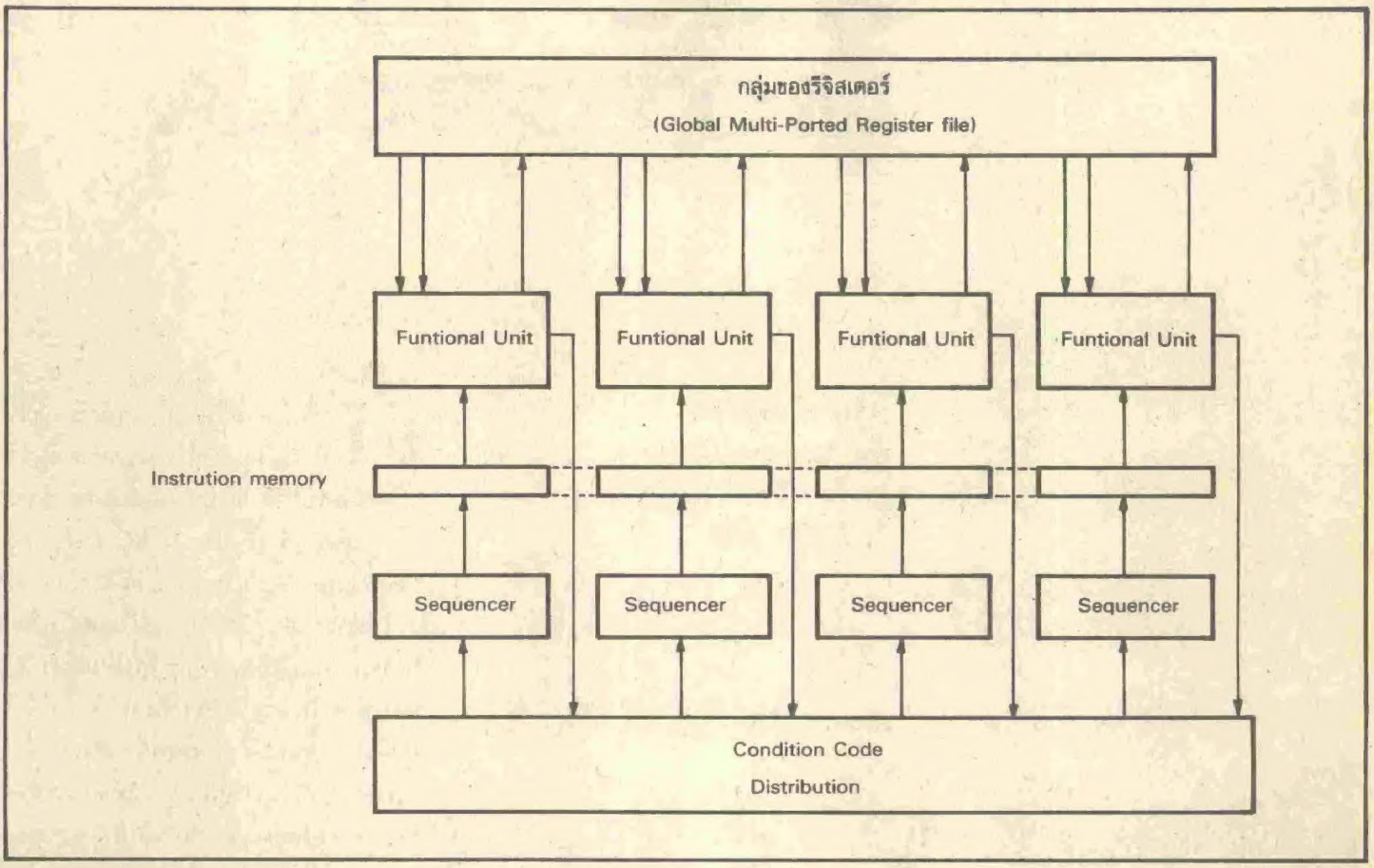
และ ณ ตำแหน่งที่มีการแยกทิศทางของการทำงาน มีคำสั่งดังนี้

$R5 \times 2 \rightarrow R5$ ถ้า $R4 > 0$

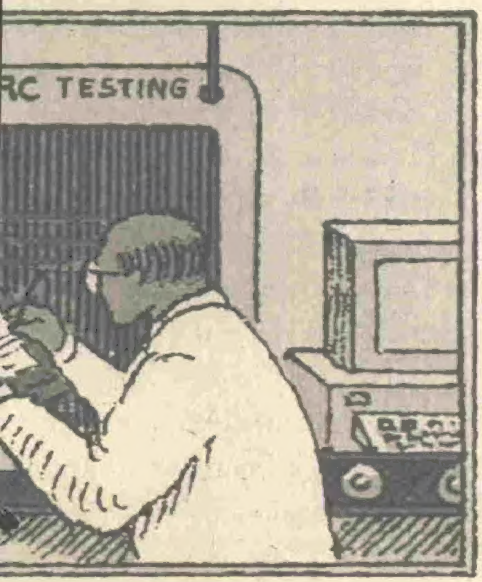
$R6 \times R6 \rightarrow R6$ ถ้า $R4 \leq 0$

เราจะสังเกตว่าทั้ง 4 คำสั่งมีความอิสระซึ่งกันและกัน แต่ถ้าซีพียูไม่ได้ออกแบบไว้สำหรับการทำงานแบบ Branch เราจะพบว่า ช่วงแรกซีพียูสามารถคำนวณ 2 คำสั่งแรกได้พร้อมกัน ในจังหวะที่สอง จะเป็นการทดสอบรีจิสเตอร์ R4 ว่ามีค่ามากกว่าศูนย์ หรือน้อยกว่าเท่ากับศูนย์ และในจังหวะที่สาม จะประมวลผลตามการทดสอบเงื่อนไขในจังหวะที่สอง ซึ่งจะเห็นว่าเราใช้ซีพียูทำงานทั้งหมด 3 ไซเคิล แต่ถ้าเราใช้สถาปัตยกรรมแบบ VLIW เราสามารถป้อนการกระทำทั้งสามได้พร้อมกัน โดยในขณะที่บวกค่า R1 กับ R1 แล้วเก็บไว้ที่ R1, บวกค่า R3 เก็บไว้ที่ R3 ก็จะมีการคำนวณ $R5 \times 2$ พร้อมกันไป และจะเก็บค่านี้ถ้า $R4 > 0$ พร้อมกันนั้นก็มีการคำนวณ $R6 \times R6$ ไปพร้อมกันด้วย ซึ่งจะเก็บค่านี้ ถ้า $R4 \leq 0$ และทิ้งผลของ $R5 \times 2$





รูปที่ 2 รูปแบบพื้นฐานของ XIMD



ไป ซึ่งจะเห็นว่าด้วยสถาปัตยกรรมแบบ VLIW นี้สามารถทำการประมวลได้ใน 1 ไซเคิลเท่านั้นเอง การประมวลผลแบบ VLIW นี้จะมีประสิทธิภาพหรือไม่มีประสิทธิภาพนั้น ขึ้นอยู่กับการแจกจ่ายงานให้กับหน่วยประมวลผลทั้ง n หน่วย ซึ่งตัวการจัดการก็คือตัวแปลภาษา (compiler) นั่นเอง ซึ่งจะเห็นว่าแม้จะมีสถาปัตยกรรมที่ดีแล้ว เราจำเป็นต้องมีตัวแปลภาษาที่ดีด้วย เราถึงเวลาที่จะต้องศึกษาด้านการแปลให้มากขึ้นกว่านี้หรือยังครับ เพราะเทคโนโลยีด้านนี้เราสามารถก้าวไปถึงได้ง่ายกว่าด้านสถาปัตยกรรมคอมพิวเตอร์ นั่นก็คือ เราควรช่วยกันสนับสนุนการค้นคว้าวิจัยในด้านซอฟต์แวร์ให้มากกว่าทุกวันนี้มิใช่หรือครับ แต่ก็ใช่ว่าจะละเลยการค้นคว้าวิจัยทางด้านฮาร์ดแวร์เสียทีเดียว สำหรับ VLIW ไม่ได้อยู่แค่ในงานวิจัย เพราะขณะนี้ซีพียูบางตัวเริ่มใช้สถาปัตยกรรมนี้แล้ว เช่น ปรมาณูชิพ (80860) แม้ว่า VLIW จะมีประโยชน์และคุณสมบัติที่ดีมาก แต่ก็ยังมีข้อบกพร่องอยู่บ้าง ซึ่งทำให้เกิดการพัฒนาและวิจัยเพิ่มขึ้น ซึ่งปัจจุบันจะเป็นเทคโนโลยีที่ชื่อ Extension to VLIW model ซึ่งชื่อว่า XIMD แล้วเราคิดว่าถึงเวลาที่เราจะต้องก้าวให้ทันกับวิวัฒนาการใหม่ ๆ หรือยังครับ